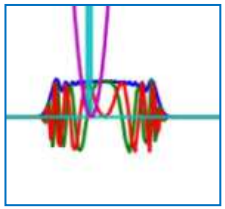
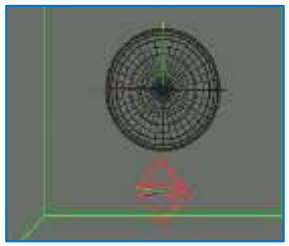
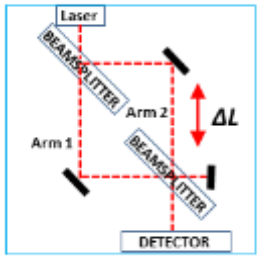
Engineering Portfolio

Morgan Rivers







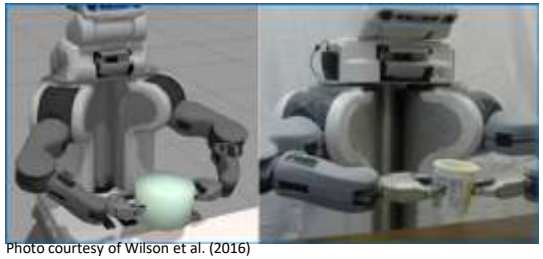


Photo courtesy of Wilson et al. (2016)

Table of Contents

About Me 3
High School Projects 4
Redesigning a Toaster Oven 5
Riemann Sum Parser 6
Simulating the Schrödinger Equation 8
Robotic Machine Learning 11
Glass Harp MIDI Controller 14
Atomic Interferometry 16
Optical Clock Rubidium Cell 18
PRNS Coherence Reducer 19
Citations 20

About Me

- I grew up in Silver Spring, MD right outside of DC and went to Wheaton High School, where I graduated from both the engineering and information technology academies. At Wheaton I was a state-qualifying cross-country runner and captain of the cross country team, placing 21st in my region. I was also captain of the trivia team "It's Academic!" for which I participated in multiple local TV competitions.
- After high school I attended Tufts University where I majored in engineering physics. Although this major is similar to applied physics, it's offered by the Tufts engineering school. I was one of two engineering physics majors in the Class of 2017 and graduated Magna Cum Laude.
- At Tufts I was a member of the early music ensemble where I learned and performed on harpsichord and portative organ. I was also president of the out in Science, Technology, Engineering and Mathematics (oSTEM) club, which advocates for queer and LGBT Tufts students in STEM fields, where I organized multiple group trips to conferences across the country.

Me at the piano

Me out climbing

- After college I went on a 3 month, 1,700 mile hike on the Pacific Crest Trail. During this time I applied to and subsequently accepted a position as an Electro/Atom Optic Development Engineer at Draper, a nonprofit R&D laboratory affiliated with MIT, after having interned there for a summer after my junior year of college.
- I then took graduate physics classes at MIT for three semesters while working at the lab, as part of the for-credit non-degree MIT Advanced Study Program (ASP).
- I now enjoy improvisatory piano performance, rock climbing, and engaging in research projects relating to global catastrophic risk reduction.

Me on the PCT at Crater Lake, OR

High School Projects

I completed a variety of cool projects before college.

- *West Point Bridge Design Competition* – Placed 7th out of a pool of thousands of virtual bridge designs built using the *West Point Bridge Design* software in 2008. The designs were evaluated based on successfully carrying a virtual moving load for the lowest cost. My bridge was the least expensive structure submitted by any middle school student.
- *Summer Intern for George Washington University Physical Chemistry Lab* – Assisted in running investigative experiments on the glass transition of amorphous solid water. Using the programming language Labtalk, I implemented an algorithm to facilitate the capture of scans from the group's Fast Scanning Calorimeter.
- *Software Developer at Heatherstone LLC (June to Aug. 2013, Dec. 2013, May to Sept. 2014)* – Worked full time on breaks from school at small startup developing large and small scale product-based and client-based projects. Primarily developed data-driven web applications. Managed four interns, overseeing and editing their code and answering programming questions. Worked with PHP, Javascript, and SQL using Wordpress and CakePHP.
  - Developed a shipping algorithm for the Sellertrax web application. It imports orders from client's Amazon seller account, determines if the product is in stock, determines how many products to ship from which locations, and communicates with the client to initiate the shipment.
  - Primary responsibility for developing an offline application in CakePHP for Rockville Fuel and Feed in Rockville, MD that analyzes raw data on hours worked and automatically generates information required for the company's employee payment program. Application is used for more than 400 employees at the company.

Fast Scanning Calorimeter[1]

Rockville Fuel and Feed

Redesigning a Toaster Oven

As part of our global product development class, our team reverse engineered and then redesigned a Black & Decker toaster oven.

Original product from Black & Decker

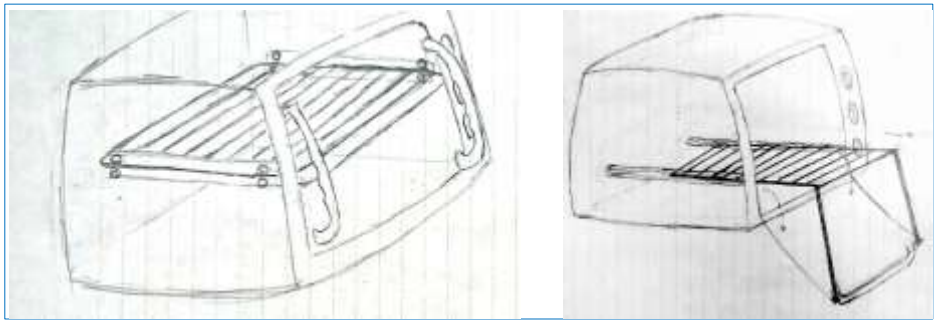
CAD model of Black & Decker product

Redesigned tray mechanism

- Myself and Arlo Clarke, Jarrett Davis, Erick Garcia, and Tyler Olney worked on the project in our Freshman year.
- I was the lead on the Solidworks modeling effort for the project because of my experience with computer aided 3D design (CAD).
- I also assisted in development of a cost-benefit analysis for the improved oven.
- The team developed a marketing strategy and a hypothetical plan for manufacture in Tianjin, China and retail in South Korea.

My Improvement Proposal

- I proposed an aluminum arm mechanism to allow the tray to automatically slide out, minimizing the chance of accidentally burning oneself when reaching into the oven.
- The solution was accepted by the group and incorporated into our CAD model, our cost benefit analysis, and our fictional marketing strategy.



Drafts of tray redesign

RiemannSum Parser

The RiemannSum equation parser was my submission for a “challenge project” in my Freshman year Applied Calculus II class.

- Challenge projects were optional assignments for extra credit issued in additional to the typical class coursework. I was the only student in the class to complete every challenge project.
- One of these challenge projects was to make a program which would calculate the left endpoint Riemann sum of any given single-variable polynomial.
- The Riemann sum is a numerical method of approximating a definite integral of any single-variable function by summing the total area of rectangles of the height of the function with a small width.
- Rather than use a package for equation parsing as was assumed by the assignment, I decided to implement an equation parsing algorithm of my own devising.
- Example Riemann sum: $\int_0^2 x^2\,dx \approx \frac{1}{50}\sum_{n=0}^{99}\frac{n^2}{2500} = 2.6268$

A zoomed-out view of the code I wrote for the project

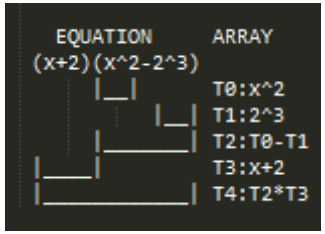
```
C:\>java RiemannSum
======Riemann Sum Calculator======
 This Program will calculate the left
endpoint Riemann sum in terms of the v
ariable x.
 Enter the n value (number of boxes)
100
Supply the lower x bound, then press [
ENTER]
0
Supply the upper x bound, then press [
ENTER]
2
Supply a function of x, then press [EN
TER]
For Example: (x^2-3*x^4+5x/(-6))\1
Allowed characters are: 1234567890.()+
-*^xX\/

x^2
width: 0.02

==============Sum===============
 Riemann Sum = 2.6268000000000034
```

The program performing a Riemann sum, with accuracy to within 10^{-14}

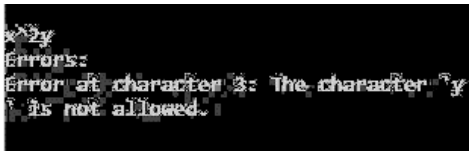
RiemannSum Parser

I'll now walk through my algorithm for parsing user input and calculating the final Riemann Sum.

- The variables, operators, numerical values, and parentheses are each their own object type which are initialized and put into an array of objects in the order of the original characters in the equation.
- To determine the order of evaluation, parenthesis groups are identified and the objects inside the parenthesis groups are parsed into tokens and added to the token list (in order of: exponent, multiply or divide, add or subtract).
  - Tokens are defined as having two values and one operation.
  - Token values can be either the variable x, a numerical constant, or another token.
  - Operations are ^,*,/,\,+, or -.
- Parenthesis groups are defined as tokens and themselves put into the token list, which can be evaluated for any particular variable value.



Example tokens for a particular polynomial



Error Detection

- Now that the equation is able to be evaluated for any particular value, the program runs through the list of operations the number of times indicated by the number of rectangles, summing all of the values from each x position and thus producing a Riemman sum.

Code snippet with a for loop *calculating the Riemann sum.* n *is the number of boxes and* y *is the area of that particular rectangle.*

```
for(int count = 1;count <= n;count++) {

    if(token_size != 0){
        for(int index = 0;index<token_size;index++){
            y = tokens.get(index).evaluate(x);//keeps evaluating each token until y is replaced by the final value
            if(first_value.signIsFlipped()){
                y = -y;
            }
        }
    }else{
        y = first_value.evaluate(x);
    }
    sum += y;

    x+=rectangle_width;
}
double riemann_sum = sum*rectangle_width;
```

Simulating the Schrödinger Equation

Before I discuss my next project, I'll introduce some physics needed for simulating the Schrödinger wave equation.

- The Time Dependent Schrödinger Equation (TDSE) describes the evolution of the complex probability density function $\psi(x,t)$, where the magnitude $|\psi(x,t)|^2$ is integrated over a range of positions x (indices j) to determine the probability of the particle being in that range of positions (indices n), given an externally derived potential energy $V(x)$. We used the TDSE with in a unit system where $\hbar = 1$ to work out a discretized solution using finite difference approximations. [2]

Explicit method

- The explicit method uses the traditional discrete TDSE:

$$(1)\ i\frac{\partial}{\partial t}\psi_n^j = -\frac{1}{2}\frac{\partial^2}{\partial x^2}\psi_n^j + V(x)\psi_n^j$$

- We can utilize the following discretizations[3]:

$$(2)\ \frac{\partial}{\partial t}\psi_n^j \approx \frac{\psi_{n+1}^j - \psi_n^j}{\Delta t};\ (3)\ \frac{\partial^2}{\partial x^2}\psi_n^j \approx \frac{\psi_n^{j+1} - 2\psi_n^j + \psi_n^{j-1}}{\Delta x^2}$$

- Plugging (2) and (3) into (1) and rearranging, we obtained ψ_{n+1} as a function of ψ_n. We can put the result into matrix form:

$$(4)\ \psi_{n+1} = (\mathbf{I} - i\Delta t\mathbf{H})\psi_n$$

- However, the ψ_n coefficient is not stable or unitary, so we did not use this method in our simulations. A unitary ψ_n coefficient will conserve net probability, so that the particle has 100% chance of being somewhere at any particular time.

Implicit method

- The implicit TDSE looks forward to n+1 to determine the rate of probability change:

$$(5)\ i\frac{\partial}{\partial t}\psi_n^j = -\frac{1}{2}\frac{\partial^2}{\partial x^2}\psi_{n+1}^j + V(x)\psi_{n+1}^j$$

- We combine this with (2) and (3), rearrange, and put into matrix form:

$$(6)\ \psi_n = (\mathbf{I} + i\Delta t\mathbf{H})\psi_{n+1}$$

- By applying an equivalent inverse matrix to the ψ_{n+1} coefficient above, we obtained a stable but non-unitary ψ_n coefficient.

Cayley Method

- The Cayley TDSE uses a centered time difference scheme[3] to determine the rate of probability change:

$$(7)\ i\frac{\partial}{\partial t}\psi_n^j = -\frac{1}{2}\frac{\partial^2}{\partial x^2}\left(\frac{\psi_n^j + \psi_{n+1}^j}{2}\right) + V(x)\left(\frac{\psi_n^j + \psi_{n+1}^j}{2}\right)$$

- We plug (2) and (3) into (7), rearrange, and convert this into matrix form:

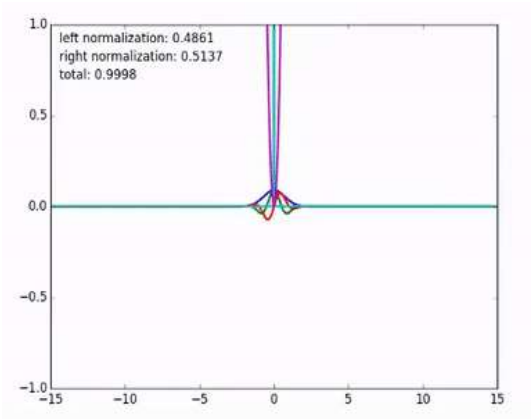
$$(8)\ \psi_{n+1} = \frac{\mathbf{I} - i\mathbf{H_c}}{\mathbf{I} + i\mathbf{H_c}}$$

where $\mathbf{H_c}$ indicates our use of a different matrix from $\mathbf{H}$. The Cayley method is an improvement to the implicit method in that that the coefficient for ψ_n is unitary.

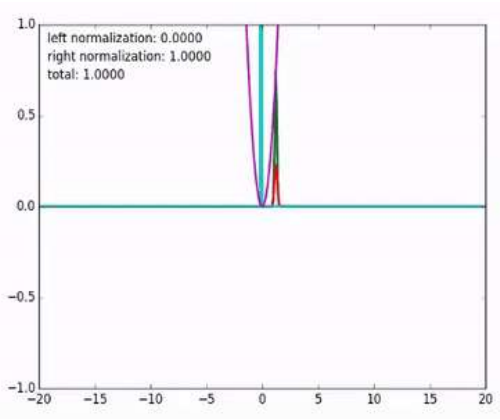
Simulating the Schrödinger Equation

Myself and physics undergraduates Ian Hunter and Ben Nissan created a numerical simulation of the TDSE for our computational physics class.

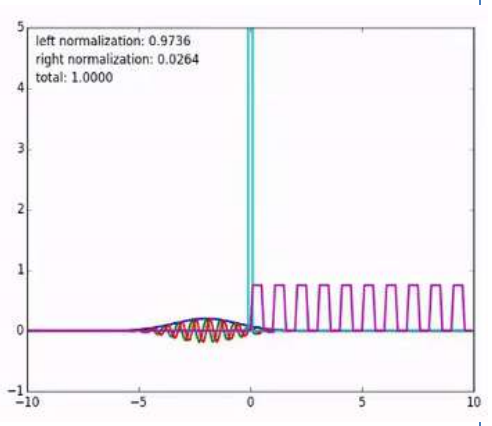
- The simulation uses a Mathematica notebook user interface, which calls our Python simulation package.
- I was primarily responsible for designing and programming the Python backend simulation package.
- The simulation could run with a bounded range in position allowing probability to leave the screen, or with a periodic range such that a travelling wave going to one end of the potential functions would loop back to the other end.
- It also could be run with either the implicit or Cayley numerical approximation methods.
- The simulation could be run with any of ten different potential functions.
- Python's FFMPEG utility was used to save animations of the simulations.



- Simple Harmonic Oscillator
- Ground state energy
- Implicit method
- Bounded



- Simple Harmonic Oscillator
- Higher Energy
- Cayley method
- Bounded



- Crystalline Potential
- Positive initial velocity
- Cayley method
- Periodic

Simulating the Schrödinger Equation

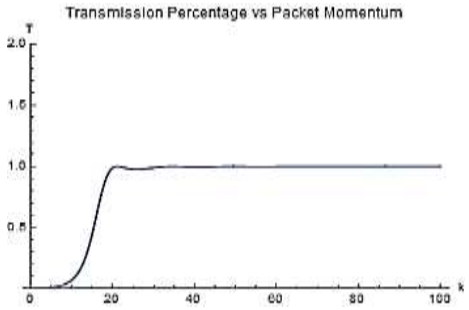
Below is a walkthrough of the algorithm I used to generate the TDSE simulations.

$$\begin{pmatrix} 2k & -k & 0 & 0 \\ -k & 2k & -k & 0 \\ 0 & -k & 2k & -k \\ 0 & 0 & -k & 2k \end{pmatrix}$$

Example matrix format for implicit bounded Hamiltonian **H**

$$\begin{pmatrix} 2k & -k & 0 & 1 \\ -k & 2k & -k & 0 \\ 0 & -k & 2k & -k \\ 1 & 0 & -k & 2k \end{pmatrix}$$

Example matrix format for implicit periodic Hamiltonian **H**



Proof it works: Cayley simulation estimates of wave packet tunneling were close to theoretical values

- I was responsible for implementing and designing the code for the matrix generation algorithm and fine-tuning the simulation parameters such that they ran smoothly and demonstrated the physics we were interested in.
- We wrote three Python classes: a matrix generation class, a potential function class, and a class that ran the simulation given the parameters passed in from the Mathematica notebook.
  - The matrix generation algorithm was customized to only calculate the matrix cells that were needed for the simulation.
    - Because the implicit and Cayley methods only use positional steps ±Δx away for the next time step, the matrix generation class only contains functions needed to calculate the diagonal elements and the cells above and below the main diagonal.
  - The potential function class took any position x and returned the value of the potential at that location.
  - The simulation runner looped through the code, incrementing the time of the simulation as it went.
    - NumPy was used to handle the complex numbers and the matrix arithmetic needed for the simulation.
    - The complex function $\psi(x)$ was calculated at each time t as a Python 1D array of complex numbers, and the value from the previous run was used to calculate the value of $\psi(x)$ for time $t+\Delta t$.

Robotic Machine Learning

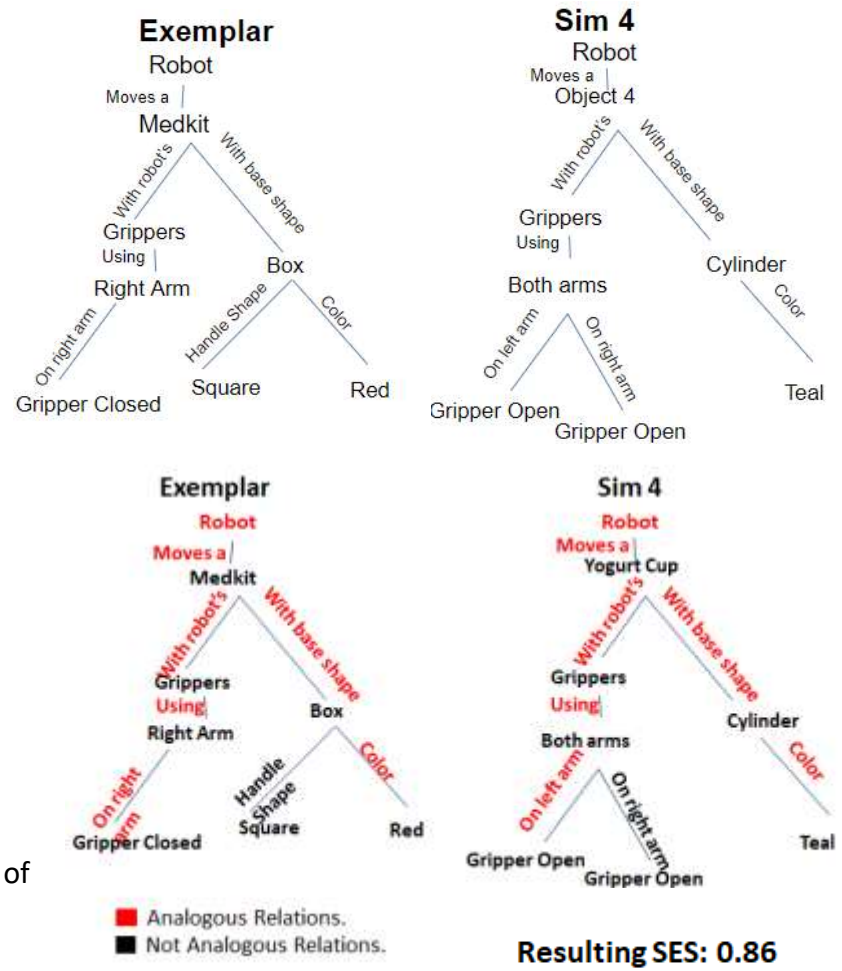
I will now explain the theory in machine learning used for robotic analogical generalization from single exemplars.

Structure Mapping Theory

- A structural mapping is any connected set of relations that holds true across multiple cases.[4]
- Advantages of structural mapping:
  - More closely resembles human rates of learning than statistical methods. [5]
  - Works even when multiple changes in variable values occur across scenarios.
  - Abstracts away insignificant variables.

Structure Mapping Engine

- Uses maximal mapping, defined as the largest structural mapping.
- Engine returns SES (Structural Evaluation Score).
  - SES = (number of relations in maximal mapping) /(total number of relations).
  - The SES corresponds to how analogous the two cases are.



Robotic Machine Learning

Publication "Generalized Robotic Action from Single Exemplars in a Robotic Architecture"[6] by myself, Matthias Scheutz, Evan Krause and Jason Wilson.

- We programmed the PR2 Robot to use the ROS operating system and ADE multi-agent system middleware with DIARC robotic interface software developed at Tufts Human Robot Interaction lab. It acts as follows:

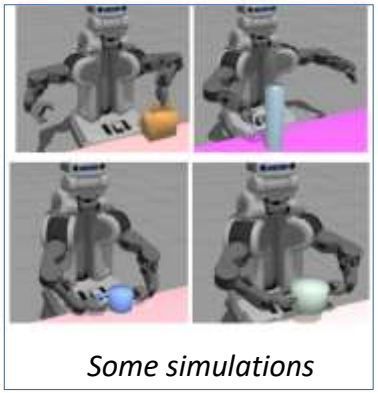
1. Robot is preprogrammed with action to accomplish task of moving an object to its right by half a meter, and performs this preprogrammed action.
2. Robot simulates series of varied task attempts in scenarios generated using Gazebo physics engine.
    - Scenarios populated with objects chosen by my variation algorithm.
3. Robot evaluates attempted actions from simulation using analogical generalization.
    - This uses an assimilation algorithm to incorporate the maximal mappings with good SES's into a generalized action framework.
    - I created the algorithm it used to evaluate the efficacy of its actions.
4. Robot attempts to accomplish analogous task using generalized action.

Algorithm 1 Generalization

```
function GENERALIZE(β, T)
    γ ←newGen(β)
    Γ ← {γ}
    for all τ ∈ T do
        flag ←F
        for all γ ∈ Γ do
            mappings ←compare( β,τ)
            bm ←greatestSES(mappings)
            ses ←score(bm)
            if ses × simulationSuccess(τ) > θ
                assimilate (τ, γ)
                flag ←T
            end if
        end for
        if flag then
            γ ←newGen(τ)
            Γ ← Γ + γ
        end if
    end for
    return Γ
end function
```

Analogical Generalization Pseudocode

Exemplar



Some simulations

Applied generalized action to new scenario

β: Base exemplar

γ: Generalization with base as exemplar

T: Counterfactual Scenarios

Γ: All generalizations

τ: Scenario

θ: Threshold

Pseudocode variable definitions

Robotic Machine Learning

Here I describe my contributions to the paper “Generalized Robotic Action from Single Exemplars in a Robotic Architecture”.

Variation Algorithm

- Effective variation for analogical generalization involves generation of a set of scene modifications spanning the configuration space.
- In my algorithm, frequent changes in childless scene variables allow inconsequential variables to be abstracted away.
- Progressively more important scene variables are changed, allowing larger scope.
- More important action variables are changed first
  - Tested riskier actions with less risky scene and then less risky actions with riskier scene.

Algorithm 2 *Variation Generation*

```
function VARY(action, scene, n_sims)
    SortByImportanceAscending(scene)
    SortByImportanceDescending(action)
    OUT ← ϕ
    scene_index ← 0
    action_index ← 0
    for i ← 1 to n_sims :
        new_scene ← scene
        new_action ← action
        SetRandom(new_scene[scene_index])
        SetRandom(new_scene[scene_index + 1])
        SetRandom(new_action[action_index])
        scene_index ← scene_index + 2
        action_index ← action_index + 1
        OUT ← OUT ∩ (new_scene, new_action)
    end for
return OUT
```

Variables Defined

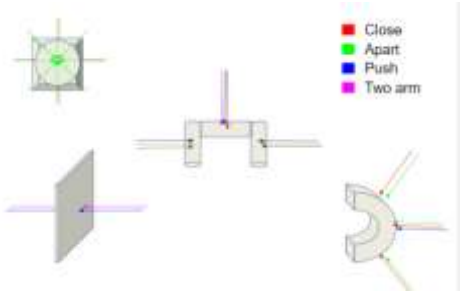
scene: list of scene variables defining the exemplar scene

action: list of action variables defining the exemplar action

n_sims: number of scenarios simulated

Implementing Rudimentary Robotic Actions

- I was in charge of programming the actions the robot used to interact with its environment, either simulated or physical.
- These low level actions had to be capable of collectively accomplishing a wide range of tasks without the use of analogical generalization.
- I allowed object grip possibilities to determine rudimentary action, and split actions into many small parts to allow more subsets of actions.
  - Handles and faces of base shapes are provided with grip points. Each grip point has allowed grip location, orientation, and “move away” distance.



Handle Grasps

Video of the robot picking a container up: https://youtu.be/hcm-nxnYd5k
Exemplar: start to 0:20. Simulation: 0:20 to 3:20. Generalized action: 3:20 to end.

Glass Harp MIDI controller

I invented a musical instrument, the “Glass Harp MIDI controller”. I’ll discuss the process which our team used to design and construct it.

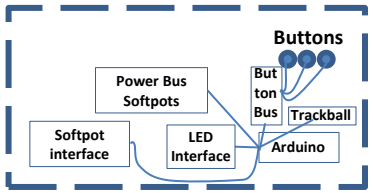
- Video on the glass harp which summarizes slides below: https://youtu.be/yLkjby2U0QE000
- The glass harp midi controller was a project from my electronic music instrument design class which I developed along with Tyler Klein, Rachel Marison, and Jackson Clawson. A glass harp is an instrument used by street performers and some professional musicians, constructed from a board with wine glasses filled with varying amounts of water to create differing pitches.

Instrument Layout and Operation

- The harp uses rotary and linear softpots for volume control and pitch selection, sustain pedals for octave selection and sustain, a trackball for pitch warbling, a knife switch to select the instrument mode, and three buttons for various sound effects specific to the mode.
- The modes are: traditional glass harp mode, Theremin mode, and a sampled mode. I worked on the glass harp and Theremin modes.

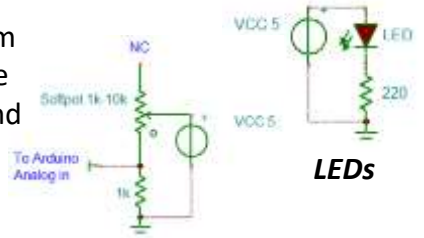
The inside of the wooden frame

Associated electronic layout

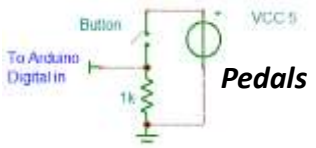


Construction and Wiring

- The instrument was constructed from a wooden frame to which holes were added for a sustain pedal box jack and a USB jack that allowed easy connecting and disconnecting from the computer.
- The acrylic top panel was first cut in half by handsaw and then brought to a laser cutter for a precision cut. The wooden board was then filed down to achieve a compression fit.



Softpots ***LEDs***



Pedals

Glass Harp MIDI controller

I will now describe the software environment and sound production techniques developed for the Glass Harp Midi Controller.

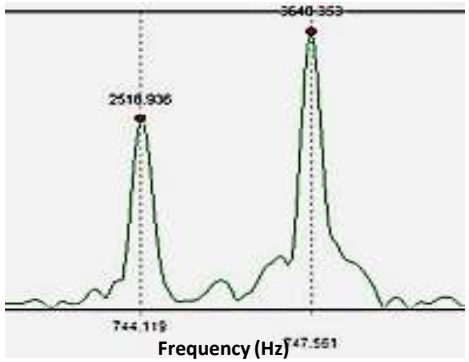
Software design

- I helped use the Max visual programming language to convert the Arduino instrument signals into MIDI format.
- The MIDI was then sent to Reason, a software package that produces sound given MIDI signals by digitally emulating hardware synthesizers, samplers, sequencers and mixers.

Arduino signals displayed in MAX

Emulating the sound of a singing wine glass

- I spent a lot of time on determining how to reproduce the sound of a wine glass. I ended up modeling the sound using a Reason subtractor with two tones 0.07 semitones apart.

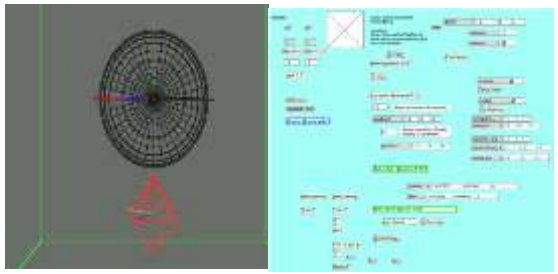


Wine glass acoustic Fourier spectrum[7]

Subtractor generating the glass harp sound

Sloshing effect

- The sloshing of water on the side of a wine glass lowers the frequency and amplitude of the sound.
- A model of the water sloshing was produced with MAX's jitter physics module which reproduces the musical effect used by glass harp players of sloshing around the water in their glasses.



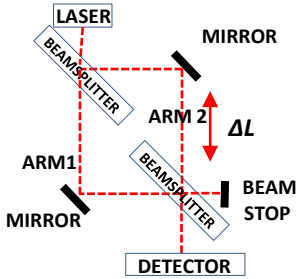
The trackball jit.phys module interface

Atomic Interferometry

Here I introduce some background for understanding cold atom atomic interferometry used for cold atom inertial sensing.

Mach-Zender Interferometers

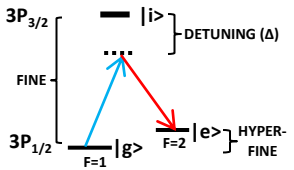
- Mach-Zender Interferometers are commonly used with laser beams to coherently split, send across two arms, and then recombine laser light.
- If the arms are different in length, the phase of the beams interfere when recombined. Mach-Zender interferometers are therefore extremely sensitive to differences in path lengths of the two arms.



Mach-Zender interferometer: As ΔL changes, optical intensity changes at detector.

Beamsplitting via Induced Rabi Oscillation

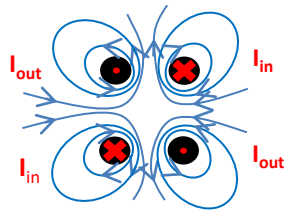
- Beams incident on cold alkali metal atoms in a vacuum with valence electrons in their ground (|g>) state can drive two photon stimulated Raman transitions.
  - These transitions utilize counterpropagating Raman beams detuned Δ from the fine structure energy splitting. The beams oscillate the atoms between their hyperfine ground state (|g>, F=1), a short-lived virtual detuned fine structure state |i>, and an excited hyperfine state (|e>, F=2) at the Rabi frequency. The virtual state is used in order to induce sufficient velocities for coherent positional atomic splitting on useful time scales[8].
- Raman beams can be pulsed some fraction of a Rabi oscillation. A pulse that is π/2 of an oscillation induces an absorption or emission 50% of the time. This is used for population splitting and recombination. A π pulse always induces either emission or absorption.



Two-photon stimulated Raman transition in Sodium

Zeeman Effect

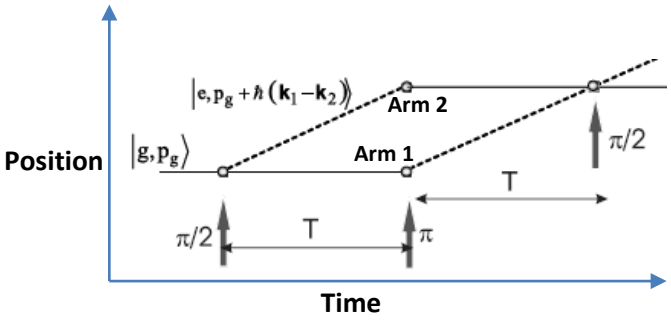
- When an atom is subjected to a magnetic field, its degenerate energy levels split. The anti-Helmholtz current configuration produces a magnetic field which can produce a spatially varying transition energy structure due to the Zeeman effect. The magneto optical trap (MOT) beams use this effect to cool and levitate atoms.



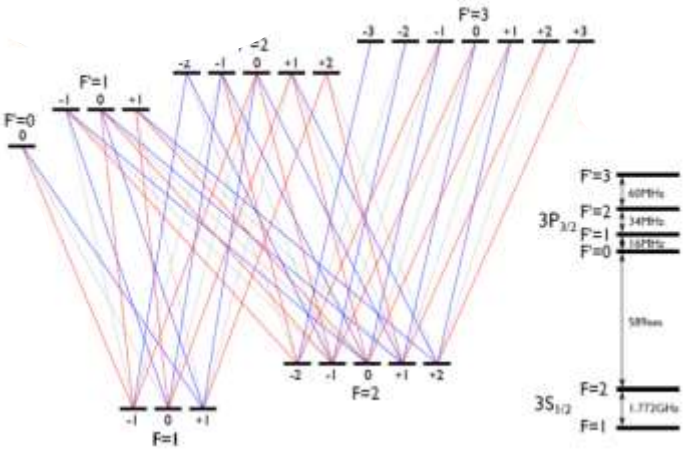
Anti-Helmholtz magnetic field

Atomic Interferometry

I'll now discuss the interferometer in "Atom Interferometry Using Stimulated Raman Transitions"[9] by Mark Kasevich and Steven Chu.



π/2-π-π/2 pulse sequence (no gravity). Note: all motion is in vertical axis. Figure taken from [8].



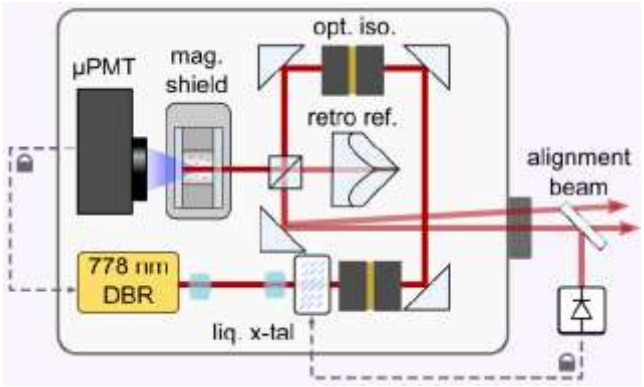
The Sodium D2 line. The Zeeman effect splits the degenerate levels shown. Red refers to the polarization of the light required for the transition from conservation of spin. Figure taken from [10].

Atomic Interferometry

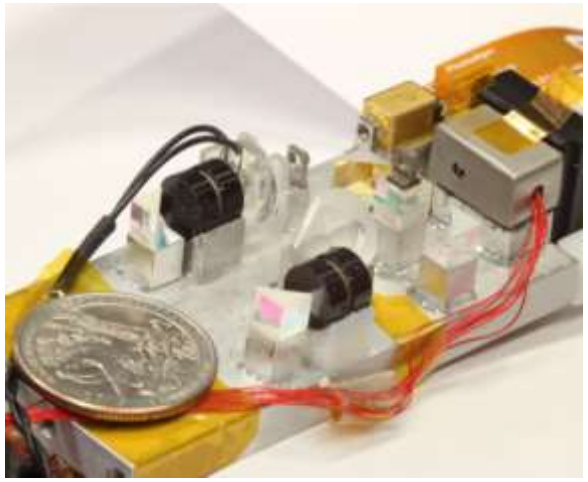
- The atomic interferometer first traps the atoms in the MOT, which also acts to cool them. The MOT is then shut off and polarization gradient optical molasses is used to adiabatically cool the atoms via stimulated emission. The atoms are then pumped into the state|g>.
- A dye laser 1.71 Ghz sideband is used to produce the Raman beams. A π/2-π-π/2 pulse sequence first uses a π/2 pulse to coherently splits the atom cloud into 50% |e> and |g>, with |g> forming a Mach-Zender Arm 1 and |e> forming a Mach-Zender Arm 2. The π pulse then flips the electronic energy states for both arms, where now Arm 1 is in the |g> state and Arm 2 is in |e>. The paths subsequently cross, and a π/2 pulse coherently recombines them. The atoms recombined to |e> are then resonantly ionized and captured in a microchannel plate.
- With no acceleration the fraction of atoms in |e> after recombination would be exactly 50% as the atoms would have no phase difference in their probability amplitudes. However, the free-evolution contribution to the phase of the atoms can be calculated from the action *S* . When the atoms are accelerated, the two arms travel different distances, causing a difference in action *S*, and hence are not phase matched on recombination.
- The fraction of atoms in |e> can be used to make extraordinarily accurate acceleration measurements.

Optical Clock Rubidium Cell

I played an important role in the construction of an exceptionally low SWAP, low tolerance optical clock



A diagram of the rubidium system. Figure taken from [11].



The atom cell we put together using my custom micro-positioning system . Figure taken from [11].

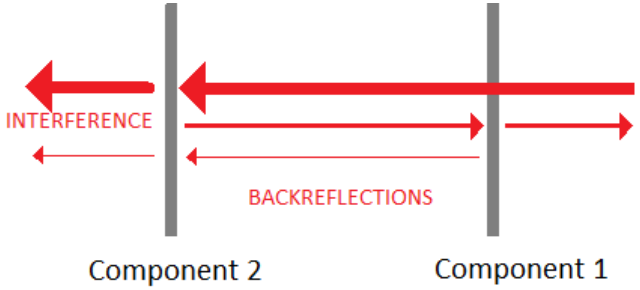
Optical Clocks

- In terms of fractional stability, optical resonances from alkali metal gases (such as rubidium) are the most stable and narrow frequency references in nature. To use this resonance, detectors monitor optical absorption of a narrow laser in the rubidium. This signal is used to lock the laser very near the narrow center frequency of the two-photon resonance. This laser is later downconverted to a countable RF frequency using nonlinear optical techniques. [11]

Rubidium Cell System Design

- With feedback from previous projects by other engineers and the help of my supervisor, I designed and constructed a repeatable, high precision robotic system that could place optical components with micron level precision on a temperature controlled aluminum baseplate.
- I also developed a proper curing system that allowed for exceptional long term mechanical stability, crucial for achieving fractional clock stability of less than $2.190 * \frac{10^{-12}}{\sqrt{\tau}}$ for averaging times of less than 10^3 seconds.
- I produced designs and tolerances for the mounts, aligned them using a micro-positioning system and set them in place with a low expansion UV-curing epoxy.
- *Unfortunately, further details on the design and construction of the micro-positioning system are not public domain.*

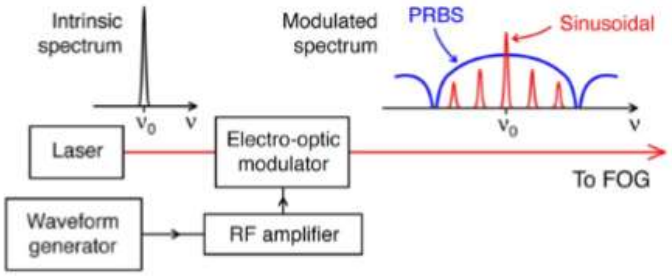
PRNS Coherence Reducer



Fabry-Perot interference due to backreflection.

PRNS for Reduced Coherence

- Narrow-bandwidth lasers tend to remain in-phase even for long propagation distances, causing unwanted intensity fluctuations at the sensor due to fabry-perot type interferences from backreflections in multicomponent optical systems.
- In [12] the authors develop a novel approach to reduce this coherence length by introducing high frequency phase fluctuations using a PRNS (Pseudo-Random Noise Source).



A diagram of the system taken from [12].

My Implementation

- I implemented the same system as the authors, but purchased higher bandwidth components with higher gains.
- Despite using a significantly more stable and more narrowband laser and than the authors, I achieved a bandwidth of roughly 40Ghz compared to the 30 GHz of broadening from their paper.
- This resulted in a coherence length reduction from about 90 meters to only 2.5 cm.

Citations

1. Sadtchenko, V. (n.d.). Vlad Sadtchenko: Associate Professor GWU Department of Chemistry. Retrieved May 28, 2018, from https://home.gwu.edu/~vlad/
2. Burke, C. J., & Atherton, T. J. (2017). Developing a project-based computational physics course grounded in expert practice. *American Journal of Physics*, *85*(4), 301-310.
3. Sehra, A. S. (2007). Finite element analysis of the Schrödinger equation. *arXiv preprint arXiv:0704.3240*.
4. D. Gentner. Structure-mapping: A theoretical framework for analogy. Cognitive science, 7:155–170, 1983.
5. A. Lovett, K. Lockwood, M. Dehghani, and K. Forbus. Modeling human-like rates of learning via analogical generalization. Analogies: Integrating Multiple Cognitive Abilities, 5:35, 2007.
6. Wilson, J. R., Krause, E., Scheutz, M., & Rivers, M. (2016, May). Analogical generalization of actions from single exemplars in a robotic architecture. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems* (pp. 1015-1023). International Foundation for Autonomous Agents and Multiagent Systems.
7. Shu Yu Yang, C. (n.d.). Wine Glass Acoustics. Retrieved May 28, 2018, from https://www.forskningsradet.no/servlet/Satellite?cid=1253965935218&pagename=VedleggPointer&target=_blank
8. Cadoret, M., De Mirandes, E., Cladé, P., Nez, F., Julien, L., Biraben, F., & Guellati-Khélifa, S. (2009). Atom interferometry based on light pulses: Application to the high precision measurement of the ratio h/m and the determination of the fine structure constant. *The European Physical Journal Special Topics*, *172*(1), 121-136.
9. Kasevich, M., & Chu, S. (1991). Atomic interferometry using stimulated Raman transitions. *Physical review letters*, *67*(2), 181.
10. Gerhardt, I. (n.d.). *Sodium D2 line transitions* [Energy Level Diagram]. Retrieved May 28, 2018, from https://gerhardt.ch/sodium/magnetic_levels_d2_01.svg
11. Maurice, V., Newman, Z. L., Dickerson, S., Rivers, M., Hsiao, J., Greene, P., ... & Johnson, C. (2020). Miniaturized optical frequency standard for next-generation portable optical clocks. *arXiv preprint arXiv:2003.13172*, from https://arxiv.org/abs/2003.13172
12. Chamoun, J., & Digonnet, M. J. (2016). Pseudo-random-bit-sequence phase modulation for reduced errors in a fiber optic gyroscope. *Optics Letters*, *41*(24), 5664-5667, from https://www.osapublishing.org/ol/abstract.cfm?uri=ol-41-24-5664